What's the Best Way to Benchmark?

As EEMBC Wrestles With Testing Conditions, Philosophical Issues Arise

A benchmark is like sex. Everybody wants it, everybody is sure of how to do it, but nobody can agree on how to compare performance.

Part of the problem lies in the fact that microprocessor performance is not a one-dimensional vector. Microprocessor drag racing is all very nice, but the average embedded designer is looking to balance the often-contradictory demands of power consumption, performance, code density, price, interrupt response, and probably other factors. A combination that's good for one application may be unusable for another.

Benchmarking embedded chips is tough, no doubt about it. That's why we have not progressed beyond Dhrystone, the accepted lowest common denominator that any microprocessor can run. Unfortunately, Dhrystone tells us very little about what a microprocessor is good at. One could argue that Dhrystone scores say more about the marketing efforts behind a chip than about its technical features.

By now you've probably heard about the embedded benchmarking work under way at EEMBC (see MPR 4/20/98, p. 13). EEMBC's laudable goal is to eradicate the scourge of Dhrystone in our lifetime. EEMBC (*www.eembc.org*) counts 24 CPU makers, large and small, among its members. For such a diverse group, they've made amazing progress toward standardizing embedded benchmarks. But there may still be some crumbs between the sheets.

Realizing that no single metric can hope to capture the many varied aspects of a chip's performance envelope, the EEMBC benchmark suite consists of dozens of smaller benchmarks. Each test contains a core algorithm taken from real-world code. There are tests for automotive-engine control, codecs, pixel manipulation, task switching, and lots of others. All the tests have been written in ANSI C for architecturally neutral portability.

EEMBC is following a path somewhat similar to that taken by SPEC (*www.specbench.org*), which is a good thing in my opinion. Specifically, EEMBC will allow its members to report two scores for every benchmark: the "out-of-the-box" score and the flat-out, fully tweaked, downhill-with-a-tailwind score. The two scores allow potential users of these chips both to evaluate competing processors under controlled conditions (the basic scores) and to see what each chip is fully capable of, given some care and attention.

Nobody disputes the need for controlled, nonnegotiable, standardized testing. But I expect some controversy over how best to handle the "tweaked" scores. Exactly how much tweaking is allowed, or desirable? Should testers be allowed to alter the source code of the benchmark? Can they rewrite key algorithms? Can they take shortcuts, like hard-coding lookup tables or—dare we say it—the predetermined results of complex calculations?

The question boils down to deciding what is important to test and what is extraneous. The Heisenberg Uncertainty Principle suggests that the less you want to know, the more accurately you can know it. If your goal is to pin down a given microprocessor's abilities in real-world situations, make sure that's what you're measuring. I believe there should be (almost) no holds barred. Any optimization, from rewriting all the C code, to creating shortcuts, to using unusual chip-specific features or instructions, is fair game in my book. This approach encourages creative and unusual solutions, which are representative of the real world of creative and unusual embedded programmers. As long as the benchmark delivers the correct answers in a reliable and repeatable manner, the details of generating the results shouldn't matter.

It's that "reliable and repeatable" part that makes people nervous. Obviously, simply hard-coding the answers to the benchmark after a few NOPs isn't meaningful. And here's where EEMBC's sister organization, the EEMBC Certification Labs (ECL; *www.embedded-benchmarks.com*), comes in. ECL must first approve every EEMBC member's benchmark scores before those scores can be published. "Approval" in this case means duplicating the same scores in ECL's own facilities. Part of ECL's role is to prove that "tweaked" scores aren't arrived at by nefarious means. That proof includes pumping alternative data sets into the chip under test to be sure that it's really executing the correct algorithm and not just regurgitating prearranged answers.

To me, it seems that a benchmark should test the abilities of a chip, not the skill of EEMBC's programmers. Wide-open testing promotes creativity and allows vendors to exploit the unusual features of their processors. As long as the chip returns the correct result under all conditions, I don't believe that what's inside the black box matters.

Forcing a particular coding convention onto dozens of different microprocessors only discourages programmer innovation and reduces everything to the lowest common denominator. And then we'd be right back to Dhrystone.